Background

Sometimes sql-servers are integrated into domain, and sometimes normal domain-users are allowed to authenticate to the sql-server. This can allow low priviliged users to gain access to sensitive data, and can also allow the user in some circumstances to be able to execute commands on the server.

It might also be possible to force the domain user that is running the database instance to authenticate.

Pre-requisites

Valid domain user

Risks

How to check for

You can enumerate SQL instances using PowerUpSQL.

```
get-sqlinstancedomain -verbose
```

However, these are based on SPNs that includes MSSQL. So there might be a lot of false positives here. For example old computer objects that doesn't really exist anymore.

You can get a list of all hosts like this:

```
get-sqlinstancedomain -verbose | select-object -property ComputerName
```

Check if your current user can authenticate to the identified mssql instance

```
$Targets = Get-SQLInstanceDomain -Verbose | Get-SQLConnectionTestThreaded -Verbose -Threads 10 | Where-Object {$_.Status -like "Acce
```

If you want to run from mssql with impacket this is the way. Also remember the `-windows-auth` flag. Otherwise it will authenticate using SQL authentication.

```
python3 mssqlclient.py -windows-auth evilcorp.local/user@192.168.1.1
```

How to exploit

Recommendation

Related Vulnerabilities

This finding should not be reported as the name of this issue, instead it should be reported as the vulnerabilities written below. The below written vulnerabilities map to the vulnerability-descriptions in that repo.

For example, if kerberoast, report as the following vulnerabilities if they are applicable:

- Weak password [ID of issue]
- Overpriviliged kerberoastable user [ID of issue]

References